7. How do you write documentation for NSE scripts using NSEDoc?

Documenting NSE scripts using **NSEDoc** is crucial for maintaining clarity and usability. NSEDoc is a built-in system for generating documentation for Nmap Scripting Engine (NSE) scripts, ensuring they are easily understood by users and developers. Here's how to write effective documentation:

1. Structure of NSEDoc

Documentation is added as comments at the beginning of the NSE script, following a structured format. These comments are processed by NSEDoc to generate a standardized HTML documentation page.

Key Sections

1. **Description**: Overview of what the script does.
2. **Categories**: Script categories (e.g., `auth`, `vuln`, `discovery`).
3. **Usage**: How the script should be executed.
4. **Arguments**: Optional or required arguments with descriptions.
5. **Output**: Expected results or output format.
6. **References**: Links to external resources or technical papers.
7. **Author**: Name and contact information of the script's creator.

2. NSEDoc Syntax and Example

Example NSEDoc Block

Here’s an example of NSEDoc documentation for a hypothetical script, `http-example.nse`:

```
--[[
Description:
This script performs an HTTP request to gather information about the target
server, including headers and content.

Categories:
- discovery
- safe

Usage:
nmap --script http-example <target>
Script Arguments:
http-example.path
  (optional string) Specifies the URL path to request. Defaults to "/".

Output:
Shows HTTP headers and the content length of the response.

Example Output:
PORT   STATE SERVICE
80/tcp open  http
| http-example:
|   Headers:
|     Server: Apache/2.4.41 (Ubuntu)
|     Content-Type: text/html
|   Content Length: 4523 bytes

References:
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers

Author:
John Doe <johndoe@example.com>
]]
```

---

3. Detailed Breakdown of Sections

a. Description

- Provide a concise explanation of the script's purpose.
- Mention its intended use case.
- Indicate whether it is safe, intrusive, or experimental.

b. Categories

- List relevant script categories:
  - `auth`, `broadcast`, `brute`, `default`, `discovery`, `dos`, `exploit`, `external`, `fuzzer`, `intrusive`, `malware`, `safe`, `version`, `vuln`.
- Example:

  ```
  Categories:
  - vuln
  - safe
  ```

c. Usage

- Explain how to run the script using Nmap.

- Include any options or prerequisites.

d. Script Arguments

- List all supported arguments, their types, and whether they are optional or required.
- Use the following format:

  ```
  Script Arguments:
  argument_name
    (data type) Description.
  ```

Example:

```
http-example.path
  (optional string) Specifies the URL path to request. Defaults to "/".
```

e. Output

- Show examples of what the script outputs during execution.
- Use indentation to indicate hierarchy if the output is structured.

Example:

```
Output:
PORT   STATE SERVICE
80/tcp open  http
| http-example:
|   Headers:
|     Server: nginx/1.21.0
|     Content-Type: text/html
|   Content Length: 1024 bytes
```

f. References

- Cite external resources or documentation that support the script.
- Include URLs or citations.

Example:

```
References:
https://developer.mozilla.org/en-US/docs/Web/HTTP
```

g. Author

- Include the name and contact information of the author.
- Example:

  ```
  Author:
  Alice Smith <alice@example.com>
  ```

---

4. Generating NSE Documentation

Once the script is written and documented:

1. **Install NSEDoc**: Ensure your system has the tools to generate documentation.
2. **Run NSEDoc**: Use Nmap's `--script-help` option or generate the HTML documentation.

Example Command:

```
nmap --script-help http-example
```

This command displays the script's documentation directly in the terminal.

---

5. Tips for Effective NSEDoc Writing

- **Be concise**: Avoid overly technical jargon.
- **Use examples**: Illustrate arguments and output with clear examples.
- **Follow standards**: Stick to the NSEDoc template for consistency.
- **Update regularly**: Keep documentation updated as the script evolves.
- **Test thoroughly**: Ensure examples and usage are accurate.

By adhering to the NSEDoc structure and providing comprehensive details, your scripts will be more accessible to the community, fostering collaboration and usability.